Aula 28/04

duas formas de atender os disp de entrada e saída. Poling de consulta, de perguntar. Não é efetivo. O conceito de interrupção é o mais utilizado em computação, quando gera uma interrupção o processador para o que tá fazendo, salva e atende a interrupção. Tem mascarável e não mascarável, depende do tipo de prioridade de interrupção que você está trabalhando. O processador tem um pino de interrupção, assim quando para ele tem um chega um sinal de interrupção ele sabe que tem algo a resolver, assim ele vai para a rotina de tratamento de interrupção, e nessa rotina, ele vai descobrir qual dispositivo gerou essa interrupção. Isso é importante porque é uma forma de comunicação do processador com os dispositivos. A técnica de pooling é eficaz em dispositivos embutidos, no caso de equipamentos no CTI que fazem a leitura dos sinais vitais do paciente, de tempos em tempos é feita a leitura e é indagado se esta tudo o.k.

Interrupção de software: ex. Um programa dividiu um numero por zero.

Interrupção de hardware:

a memoria RAM armazena os processos que estão sendo executados no momentos.

Hierarquia de memoria: o que esta em uso no momentos está na cache,

cache, processadores e RAM são voláteis e os disco rígidos não voláteis.

Exemplificando: a biblioteca seria o disco rígido, a cache são os livros abertos que estão sendo usados no momento e os livros escolhidos ficam na RAM enquanto não são usados pelo cache.

Qualquer dispositivo de entrada e saída tem sempre que passar pelo sist. operacional. Fazer um programa de looping infinito, consome muita cpu, mas não consome toda a cpu, pois após um tempo que o programa está em execução, ele vai ser retirado da execução para colocar outro que esta na fila, logo o SO não deixa ele dominar o processamento 100% do tempo em looping infinito.

Estruturas

processo é um programa em execução.

Sistema operacional é responsável por criar/remover qualquer processo que está rodando.

Quando um processo vai comunicar com outro processo, o sistema operacional tem que atuar gerenciamento da memoria principal é feito pelo sistema operacional. Gerencia espaço vazio, espaço usado, verifica se o endereço pertence ao processo que está sendo executado, se não for ele mata o processo. Mantêm informação sobre quais partes da memória está sendo usada.

O sist op gerencia os arquivos, tem que saber os que estão sendo usados e como estão sendo usados. Quando se apaga um arquivo do HD ele não é excluído, para excluir permanentemente um arquivo, é necessário zerar os blocos do HD. Tem programas que fazem essa função, ou seja, preenche por cima de todo o espaço do HD, assim após isso os arquivos que devem ser excluídos assim serão feitos.

No sistema operacional temos que ter um sistema de proteção de tudo que é usado, ou seja, memoria, processador e etc. Ou seja não podemos deixar um programa fazer uma operação que possa danificar os hardware.

O shell é o interpretador de comandos, ele serve para comunicar direto com o núcleo do sistema operacional.

O sistema operacional que vai tomar conta da alocação de memória, podemos por exemplo ver quanto de memoria que cada usuário está consumindo.

Como o programa executa uma operação privilegiada, como o printf, que acessa ao monitor, se o programa não pode acessar o monitor o sistema operacional vai ter que utilizar o monitor, logo o temos que “avisar” o sist operacional que ele deve usar o monitor. E um conceito chamado de chamada ao sistema ou system call. É uma função, que vai realizar uma chamada ao sistema, o programa não executa a função, e sim o sistema operacional executa a função, toda chamada privilegiada deve ser feita assim, o sistema operacional que vai realizar a execução para poder acessar hardware.

Aula 05/05

quando o processo ocupa uma posição ele vai ocupar um tamanho,
a memoria é como se fosse um vetor, é dividido em um segmento de código onde o código é compilado e copiado. A stack armazena as variáveis locais, na conversão e trams,ossao o valor mais significativo fica primeiro. É convertido em um determinado tipo de padrão para facilitar o entendimento. Lembrar do exemplo de duas pessoas de nacionalidade diferentes conversando em inglês que é uma linguagem em comum.
Armazenamento estático.
Temos o espaço de endereçamento é a memoria que o processo pode acessar. Temos que armazenar o estado do processador, para quando for executar outro processo vou parar num ponto onde estava.
O que é um contador de programa: é um registrador que armazena a próxima instrução a ser executada, para saber qual a próxima que irá executar, fica no espaço de endereçamento do processo.
O processo não é inteiramente carregado na memória principal. Quando a memória está cheia o SO tira da memória principal e manda pro HD.
Process id é o número identificador do processo atribuído pelo SO.
Userid é o número de usuário no SO.
Todo vez que criamos um processo o SO vai preencher alguns dados e irá ocupar uma quantidade de memória para rodar o software.
Chaveamento de contexto estamos trocando de um processo para outro.
Computadores que possuem mais núcleos tem vantagem pois podem executar mais de um processo ao mesmo tempo.
Criar um processo filho é basicamente ter um programa e criar um filho para usar no programa que você está executando, assim os ois vai entrar na fila de execução, pro código o filho pode ser executado em momentos diferentes durante o programa, ele não depende da execução do pai para ser exeutado. **O pai e filho só disputam o processador, pois se estão em áreas de memoria diferentes eles não compartilham as variáveis.**
Temos 3 categorias de escalonadores: curto prazo, médio prazo e longo prazo. Longo prazo é executado quando o processo é criado, de curto prazo é usado na chamada ao sistema, i/o, etc. Médio prazo determina quais vão ficar ou sair da memória.
Escalonamento chamado de swap, quando os processos dependem mais de entrada e saída do que de cpu. Ou seja, está parcialmente na memória.
O objetivo do sistema operacional permitir a multiprogramação, executar a maior quantidade de processos e a cpu ficar ociosa.
O processo pai cria um filho, aí e parecido com árvore. O processo pai tem o PID quando cria o filho se cria o PID do filho, PPID vai ser o identificador do pai. Eles são ligados e o filho pode herdar algumas variáveis do pai, porém eles ficam em partes distintas na memória. PID do filho = 0, coloca o código que o filho vai executar, no else, coloca-se o comando que o pai vai executar pois se o do filho for diferente de zero, iremos executar o comando do pai. Pode acontecer de processos que viram zumbi, ou seja, consome CPU sem produzir nada. O filho quando fica sem o processo pai, pode virar um processo zumbi se não for morto.
Treads é uma estrutura que permite executar diversas funções ao mesmo tempo. Usando mais núcleos, o uso das treads é uma boa escolha quando se fala em desenvolvimento e execução de programas. A tread também é chamada de processo leve. Podemos criar treads específicas também, como por exemplo com alguma função específica como por exemplo o uso de pilhas.
Se mais de uma tread tenta acessar a mesma variável ao mesmo tempo da escrita. Podemos ter um conflito.
Para proteger os recursos compartilhados, utilizamos os semáforos parei é usada para o controle de variáveis, as treads o próprio S.O. faz o controle. Cada tread pode executar um pedaço diferente do programa. Temos também o bloco de controle de processo com as treads associadas a ele. Pode ter tread pro núcleo gerenciada pelo S.O. e também podemos ter no espaço do usuário.

Ptread create + nome da função, ao passar esse código, ele vai executar a função em uma tread. No sistema web, para cada requisição ele vai criando uma tread. Quem aloca as funções nas treads e núcleos é o sistema operacional, não tem como direcionar isso, o próprio SO define.
A tarefa menor vai ser executada primeiro no FIFO, com isso reduzimos o tempo de espera para executar os outros.

Aula 19-05
No chaveamento de contexto, a tread começa a ser executada mas pode ser interrompida, a variavel x tem valor igual a 4. Por algum motivo a tread 1 vai pegar a variável x e passar igual a 5nesse momento, ocorre um chaveamento de contexto, ai a tread 2 le o valor, ela vai ler o valor 5 se terminou de escrever e vai ler 4 se a outra não terminou a operação de escrita. O problema é não conseguir determinar o que vai ocorrer. Para solucionar temos que pegar as variáveis que são compartilhadas, vamos ter que criar um mecanismo de proteção, criar um semáforo, pois vai ter uma região compartilhada entre as treads, ou seja, toda a area compartilhada tem que ter proteção, na computação o semáforo é uma ferramenta de sincronização de processos. Quando nenhuma tread está usando a área do semáforo dizemos que ele tem o valor =1. Quando s for menor ou igual a zero a tread fica parada, quando a variável sai do semáforo, ele incrementa o s, logo sai do loop de menor igual a zero. Essa variavel S é compartilhada em duas ou mais treads, a tread 1 vai tentar acessar a variável compartilhada, se ela acessa o S no inicio vai ser igual a 1, enquanto s for menor ou igual a zero ela fica parada, porem o s é igual a 1, então a tread decrementa o s, o s vai igual a 0. ai entra na região critica que é a região compartilhada. A tread 2 tenta entrar na região critica, o S é igual a 0, logo a tread vai ficar parada
“se s for maior que zero, região crítica.
A função de liberação do semáforo incrementa o s, a função do loop decrementa o s, aí fecha o semáforo e ninguém entra.
Quando s = 1 porta aberta, no momento que entra, s=0, logo se outra tread tentar entra na porta o s= 0 a porta está fechada. Quando sai da região crítica, a tread incrementa o s, s=1, a outra tread pode entrar porque s =1.

região crítica é uma região compartilhada, tem que disciplinar o acesso aquela área.
Região compartilhada é uma estrutura de dados, onde vária treads podem acessar a variável
semáforo de contagem é um semáforo tem outros valores alem do 0 e 1, logo entrariam mais treads na região crítica.
Toda vez que entra no semáforo os processos vão sendo enfileirados, se o S for menor que zero, ele entra em fila para aguardar seu incremento vai ser retirado da fila e sera incrementado, aí pode executar.
Buffer cheio não =

buffer vazio

Jantar de filósofos, cada filosofo é uma tread, o problema é como colocar os semáforos, para permitir que todos os filósofo possam comer. Problema classico de sincronização. O filoosofo 1 espera o talher da direita e depois espera o da direita, come e libera os talheres.
Soluções: Não espere pelo talher, pegue o da direita e tenta pegar oda esquerda, se não conseguir, libere o talher que pegou.
Tentamos pegar os talheres em ordens aleatórias.
Deixa rodar sem se preocupar com nada.

Impasse, parei 1:13:00